1.0 RATIONALE.

1.1 The Original BASIC ENVIRONMENT

"...the choice of monitor commands, such as HELLO, NEW, OLD, SAVE, LIST, RUN, and GOODBYE. Short common English words as system commands encouraged computer use by nonexperts. The true impact of this choice is almost beyond comprehension to a computer specialist, who is quite at home with 'logon', 'catalog', 'execute', and 'logoff', or their single letter abbreviations that are so common. Persons who use BASIC but don't understand the distinction between a language processor and an operating system monitor are incredulous when they learn that NEW, OLD, and RUN are not part and parcel of BASIC. The moral is obvious: simple and understandable command languages, like simple programming languages, are essential if nonexperts are to use the computer." [KURTZ 78] p.107

1.2 Guiding Principles

1. Keep it simple;
2. If it's difficult to explain then the design is faulty;
3. It must be a well-engineered system;
4. Single level command structure;
5. COMAL is primarily an interactive programming language;
6. A conceptual model must be presented to the user -- otherwise the user will invent his own!

1.3 Surface Syntax

The phrase "surface syntax" is used to denote the structure of a language in its written form. Man-machine (man-computer) dialogue is carried out primarily by means of the written word (command and response languages, programming languages, etc.,). Therefore, the surface syntax of the language chosen for such man-machine communication is of great importance from a human engineering point of view. [LEDGARD 81]

1.3.1 The Programmers' Jargon - Every (new) scientific discipline generates its own jargon (specially coined words and phrases) to convey the meaning of new concepts. Computer Science is no exception. Newcomers to the discipline are expected to adopt the jargon and to make it a part of their working vocabulary. In Computer Science there is a growing body of casual users (i.e. non-programmers) who will not accept the current jargon in its present form. There is, as a result, a real need to determine what is "essential computer jargon".

1.3.2 Natural Language As A Model - The surface syntax of the command language ought to resemble that of the natural language of the user. Firstly, this implies that the overall structure of the command statement be:

<command verb <object_phrase

EXAMPLE
One ought to use:

```
LIST 10,20
```

and not

```
10,20 LIST
```

Secondly, the <command verbs ought to have a meaning closely related to their normal (English) usage. EXAMPLE
One ought to use:

```
COPY TO myfile
```

and not

```
SAVE myfile
```

1.3.3 The Workspace - The WORKSPACE is that area of the system wherein the user builds his own programs. The WORKSPACE is partitioned into slots, numbered 0 to 9999. Each slot holds one program line. To enter a program the user indicates his intention by issuing the **PROGRAM** command, optionally followed by a program name. This name is associated with the program throughout its lifetime. In the event that the user does not give a program name, the system will prompt for this information at the end of the current program definition session.

R E F E R E N C E S

1. [KURTZ 78]
Kurtz, Thomas E.
"BASIC"
ACM SIGPLAN Hist. of Prog. Lang. Conf.
SIGPLAN NOTICES Vol.13 No.8 (August 1978)

2. [LEDGARD 81]
Ledgard, Henry., Singer, Andrew. &
Whiteside, John.
"Directions in Human Factors for Interactive Systems"
Lecture Notes in Computer Science Vol.103
Springer Verlag (1981)

2.0 COMMAND LANGUAGE DEFINITON.

The COMAL commands are as follows:

ASSIGN	COPY	DELETE
DISCARD	EXECUTE	EXIT
EXPLAIN	LIST	PRINT
PROGRAM	QUERY	RENAME
RESUME	SHOW	USE
INSERT	NUMBER	DATA

2.1 ASSIGN COMMAND

```
assign_command :
        variable := expression
    ;
```

The **ASSIGN** command is used for simple 'debugging'. The variables occurring on the left-/right-hand-side of := are to be found in the SYMBOL TABLE. EXAMPLES

1. COMAL A := SQRT(C)

```
        -- assign the value of SQRT(C) to the variable
    A
```

2.2 COPY COMMAND

```
copy_command :
          COPY FROM source
        | COPY TO target
        | COPY FROM source TO target
        ;
```

EXAMPLES

1. COMALCOPY FROM myfile

   -- load the contents of "myfile" into the WORKSPACE;

2. COMALCOPY TO myfile

   -- save the contents of the WORKSPACE in "myfile";

3. COMALCOPY FROM thisfile TO thatfile

4. COMALCOPY FROM SCREEN TO PRINTER

   -- used to obtain a 'hardcopy' of the image on the SCREEN;

2.3 DATA COMMAND

```
data_command :
          DATA filename
          ;
```

The DATA command is used to enter data into the workspace, so that the user can create ASCII files.

2.4 DELETE COMMAND

```
delete_command :
          DELETE
        | DELETE  source
        | DELETE  range
        ;
```

EXAMPLES

1. COMALDELETE

   -- delete the contents of the WORKSPACE and SYMBOL TABLE; does not take effect without an affirmative response to challenge;

2. COMALDELETE myfile

   -- delete the contents of "myfile" and remove the name from the user's environment;

3. COMALDELETE 50 500

   -- delete lines 50 to 500 of the user workspace;

2.5 DISCARD COMMAND

```
discard_command ;
        DISCARD
      | DISCARD package_name
      ;
```

EXAMPLES

1. COMALDISCARD

   -- discards all packages,after a positive user response to a challenge;

2. COMALDISCARD TURTLE_GRAPHICS

   -- discard the TURTLE_GRAPHICS package from the user's working environment;

2.6 EXECUTE COMMAND

```
execute_command ;
        EXECUTE
      | EXECUTE program_name
      ;
```

EXAMPLES

1. COMALEXECUTE

```
-- run the program currently in the WORKSPACE;
```

2. COMALEXECUTE program_name

```
-- locate the program "program_name";if it is
not in the WORKSPACE then COPY it FROM the
FILESTORE TO the WORKSPACE and EXECUTE it;
```

2.7 EXIT COMMAND

```
exit_command :
        EXIT
      | EXIT QUICKLY
      ;
```

The **EXIT** command is used to leave the system. Normally, the system will carry out certain tests on the state of the user's files etc., and report back before logging the user off the system. The system may request whether the user wants to save the current version of a particular program resident in the workspace etc. If the user does not care then he may **EXIT QUICKLY**.

2.8 EXPLAIN COMMAND

```
explain_command :
        EXPLAIN object
      ;
```

EXAMPLES

1. COMALEXPLAIN LIST

2. COMALEXPLAIN SYMBOL TABLE

The EXPLAIN command requests further information on any of the objects in the COMAL system.

2.9 INSERT COMMAND

```
insert_command :
          INSERT [ AFTER ] linenum increment
          |INSERT source [ AFTER ] linenum
          ;
```

The INSERT command allows the insertion of pre-written or user-generated text into the current workspace. Automatic line renumbering takes place if the text is taken from a file source.

2.10 LIST COMMAND

```
list_command :
          LIST [ range ]
        | LIST [ range ] source
        | LIST  [ range ] target
        | LIST [ range ] source ON target
        ;
```

EXAMPLES

1. COMALLIST

```
        -- list the contents of the WORKSPACE on the
    SCREEN;
```

2. COMALLIST ON PRINTER

```
        -- list the contents of the WORKSPACE on the
    PRINTER;
```

3. COMALLIST myfile ON PRINTER

```
        -- list the contents of "myfile" on the
    PRINTER;
```

2.11 NUMBER COMMAND

```
number_command:
          NUMBER linenum increment
          ;
```

The NUMBER command renumbers the users entire workspace.

2.12 PRINT COMMAND

```
print_command :
        PRINT variable list
      ;
```

EXAMPLES

1. COMALPRINT A

```
        -- print the current value of the variable A
```

2.13 PROGRAM COMMAND

```
program_command :
        PROGRAM
      | PROGRAM program_name
      ;
```

The **PROGRAM** command is used to enter program text into the WORKSPACE. The system provides automatic line numbering facilities (starting at 10, with increments of 10). The user escapes from the program definition mode by using the <ESC key. Resumption of program definition mode is possible by using the **RESUME** command. When the user issues the **PROGRAM** command the system **DELETE**s the current WORKSPACE and SYMBOL TABLE. EXAMPLES

1. COMALPROGRAM

```
        -- enter program definition mode; the program
    name will be supplied later;
```

2. COMALPROGRAM program_name

```
        -- enter program definition mode; the program
    name is "program_name";
```

2.14 QUERY COMMAND

query_command :
?
;

EXAMPLES

1. COMAL?

-- this is a cry for HELP; the response from the system depends on its sophistication;

2. COMALPROGRAM ?

-- the system will prompt for the program name

2.15 RENAME COMMAND

rename_command :
RENAME old_name TO BE new_name

The purpose of this command is to provide a flexible means whereby the user can 'tailor' his own environment. The RENAME command is essentially a 'string manipulation' command. EXAMPLES

1. COMALRENAME WORKSPACE TO BE WORK AREA

2. COMALRENAME thisfile TO BE thatfile

2.16 RESUME COMMAND

resume_command :
RESUME
;

EXAMPLES

1. COMALRESUME

-- resume the execution of the currently halted job;

2.17 SHOW COMMAND

```
show_command :
          SHOW ALL
        | SHOW objects
        | SHOW object OF packagename
        ;
```

EXAMPLES

1. COMALSHOW COMMANDS

2. COMALSHOW COMMANDS OF GRAPHICS

   -- shows what commands are available in the GRAPHICS package;

3. COMALSHOW FILES

4. COMALSHOW PACKAGES

5. COMALSHOW PROGRAMS

6. COMALSHOW SIZE OF CALPACK

   -- shows the memory required by the package CALPACK;

2.18 USE COMMAND

```
use_command :
          USE package_name
        ;
```

EXAMPLES

1. COMALUSE EDITOR

   -- EDITOR is a standard system-supplied package for editing files;

2. COMALUSE GRAPHICS

   -- GRAPHICS is a standard system-supplied package for producing vector (raster ?) graphics.

3. COMALUSE PACKAGE_BUILDER

   -- PACKAGE_BUILDER is 'the' standard system-supplied package for building user-defined

packages;